

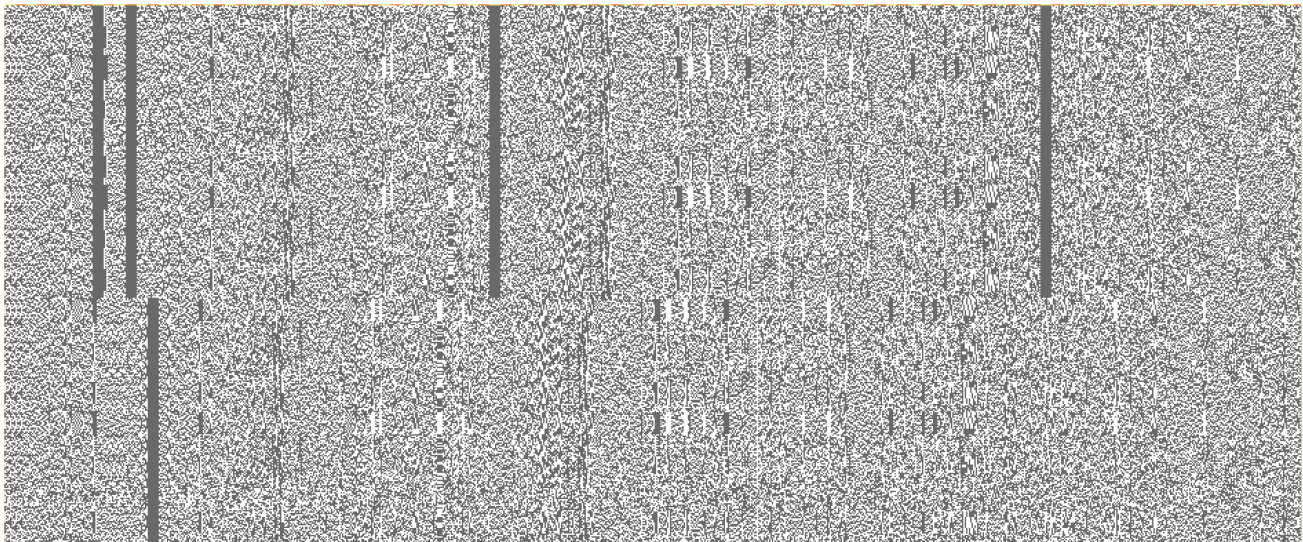


NAND Bad Columns. Bad Column Remover

Each time during the chip-off data recovery, after the physical image extraction, and before any further step, it's necessary to check the quality of the dump. Due that memory chips are being read physically, the extracted dump may contain defects. In some cases, we can observe that one physical plane is damaged, in other cases, we have a lot of bit errors in the dump. And also, we can distinguish another type of defect that is common for modern NAND memories, these are bad columns. Bad columns or bad bytes are places where a plane of NAND memory crystal is physically damaged/breached and data can't be written and stored on these particular offsets/bytes.

Before any data will be written to the NAND, the controller is analyzing memory for quality, among the others, the controller is checking if bad columns are present. To do that, the controller sends a special command and, as an output, gets the positions of defective bytes. Afterward, during data recording, the controller is filling out positions where bad columns are presented with some hex values which are usually static and not related to the recorded data. A similar situation takes place during the reading of data from the NAND memory, when the controller is sending the read page command, the whole physical page is loaded to the controller buffer, and then, in the buffer, bad bytes are skipped.

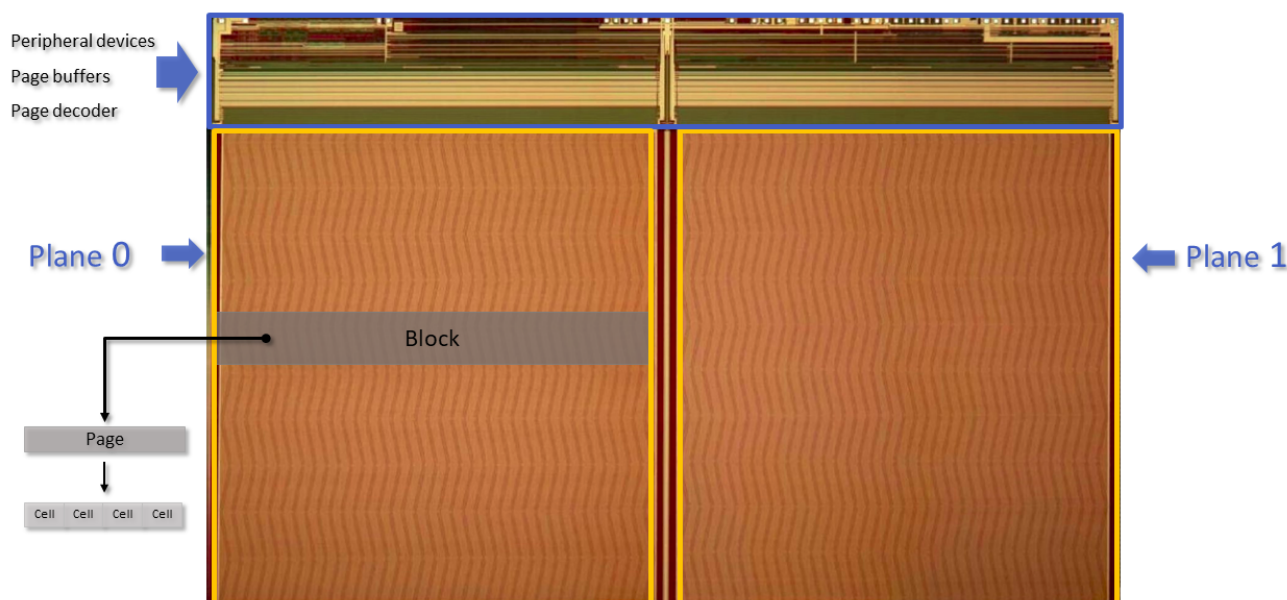
Example of bad columns via VNR Bitmap viewer



When the controller is recording user data to a memory chip, it has to select the crystal (CE) to which data will be sent, then it sends the page program command and starts data transmission. Typically memory chip consists of 1/2/4 or 8 identical crystals. Memory crystal, as a whole, isn't a unit where just user

data is stored permanently, it consists of peripheral devices, page buffers, page decoders, and planes. Physical planes are arrays of memory cells that are composed into pages and blocks and this is the exact place where user data is written and stored. In the same way as memory chips, NAND crystals may consist of single or multiple planes. The most popular is 2 planes, but we can distinguish also crystals with a single physical plane, 4,8, or even 16 planes.

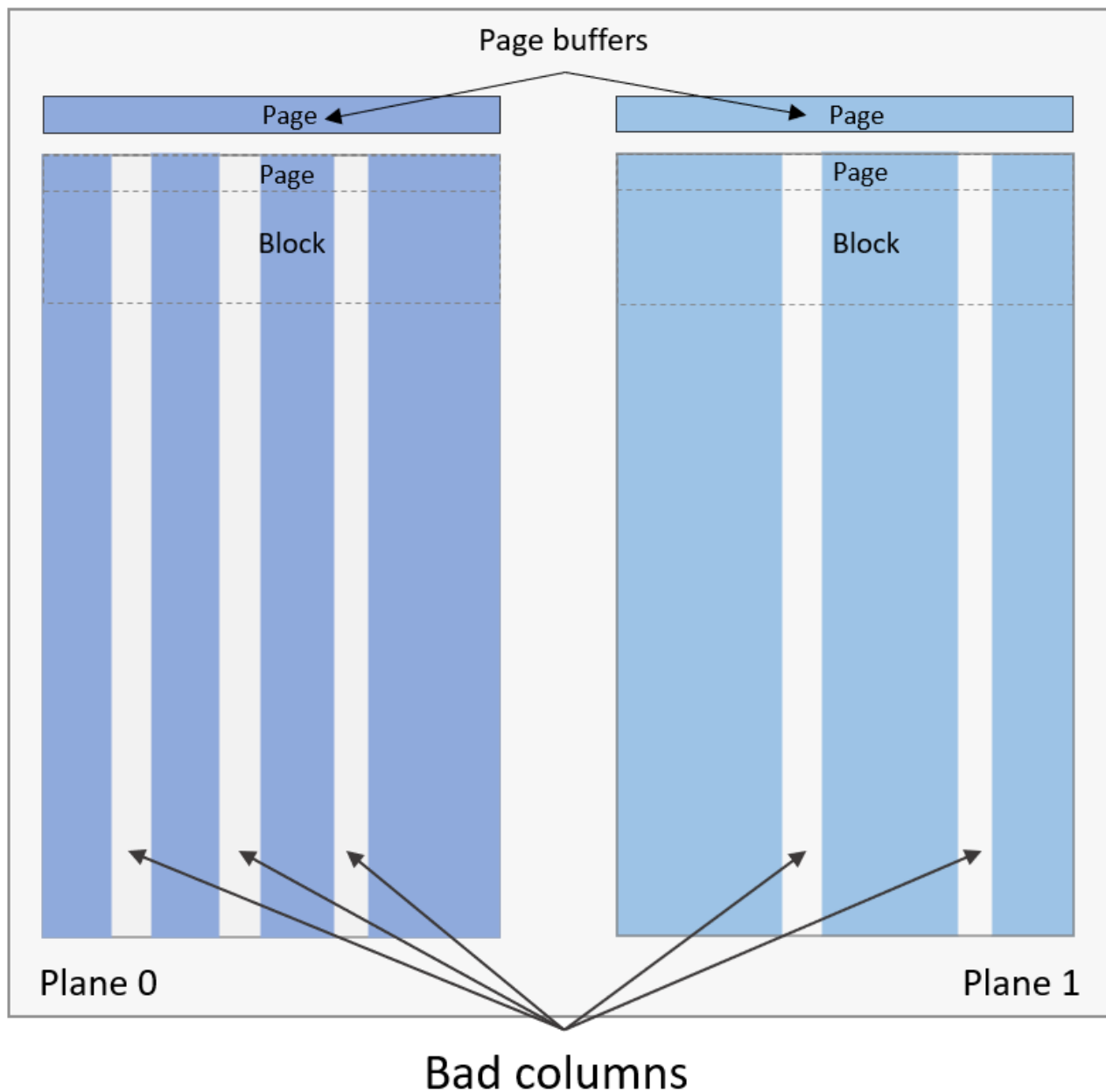
Decapsulated NAND memory chip



Modern NAND memories, especially TLC chips, have relatively poor quality, therefore quite often they contain some factory defects. These defects are individual for each crystal inside the NAND memory chip and for each plane inside the crystal. It means that for each plane, inside the crystal, bad columns may present in different positions, and for each plane amount of them can be different.

Bad columns inside the NAND memory Crystal (CE)

In the presented example, we have a single crystal with two planes where there are bad columns. Each plane in crystals of NAND memory consists of hundreds of blocks, and each block consists of multiple pages. Bad columns are affecting the whole plane in a particular position, so all blocks and pages inside the plane will be damaged on this exact position/byte.



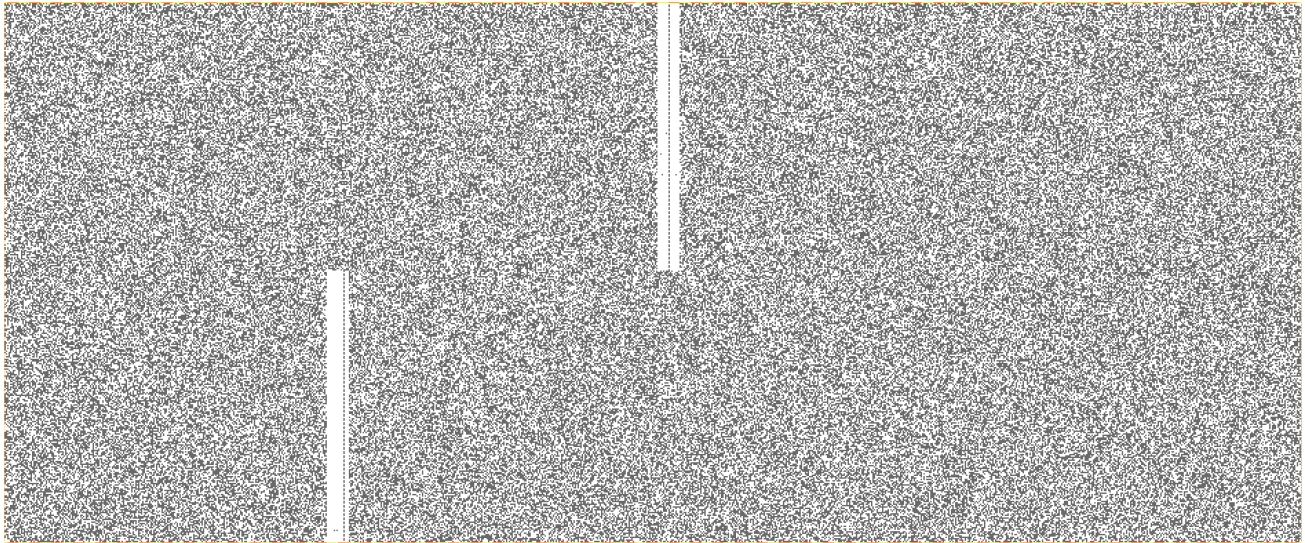
For each plane inside the NAND memory crystal amount of bad columns can vary. Usually, in cases where bad columns are present, we may have around 80 of them. In some severely damaged chips, the number of defects can be significantly higher, sometimes the number of bad columns can oscillate even around 300, for each plane. Typically the size of bad columns is 2 bytes, however, we can also distinguish 1-byte bad columns, which are occurring rather rarely. In some cases, multiple bad columns can be grouped together, so as a result, you may spot a bad column that has 6 bytes and sometimes around 20 or even more.

Types of bad columns

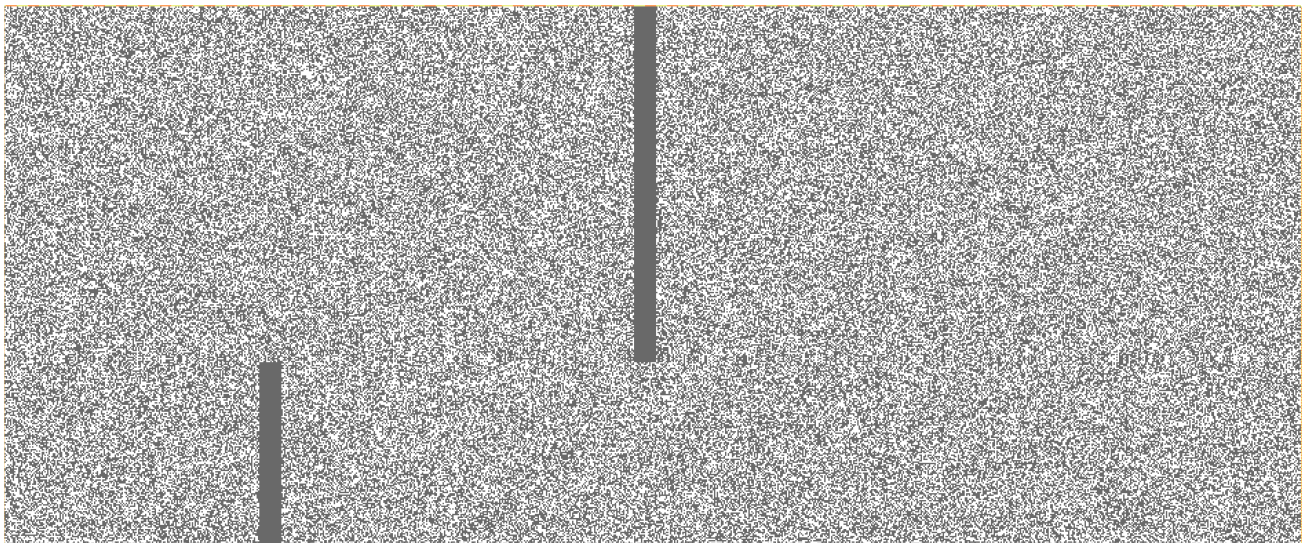
When a controller is preparing a page to send to NAND memory, positions of bad columns are being filled up with some HEX values. In many cases, these values are static and very distinctive compared to the other recorded structures, like ECC areas or especially Data Areas. The pattern of bad columns depends on the controller model and vendor, however, in the majority of cases

bad columns have two typical binary patterns. The most popular are 0xFF and 0x00. In some specific cases this pattern can be slightly different, but usually is static and not related to a type of data recorded by the controller.

Ex.1 0x00 - Bad column



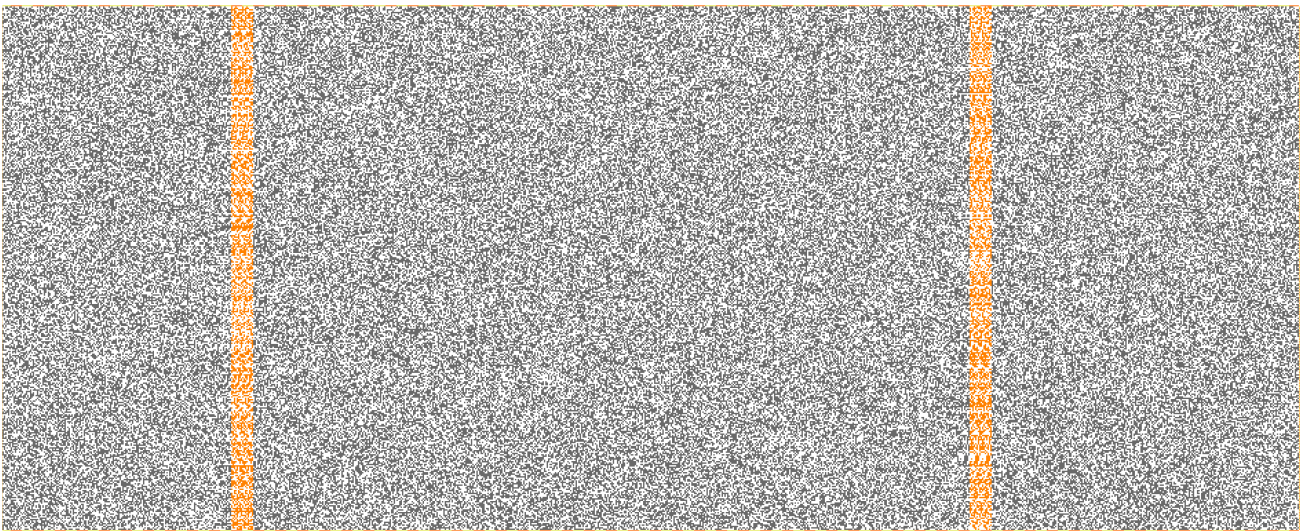
Ex.2 0xFF - Bad column



Both these, mentioned above, examples are typical for majority controllers, like Silicon Motion, Phison, Toshiba/SSS, Sandisk, and for some Alcor Micro as well.

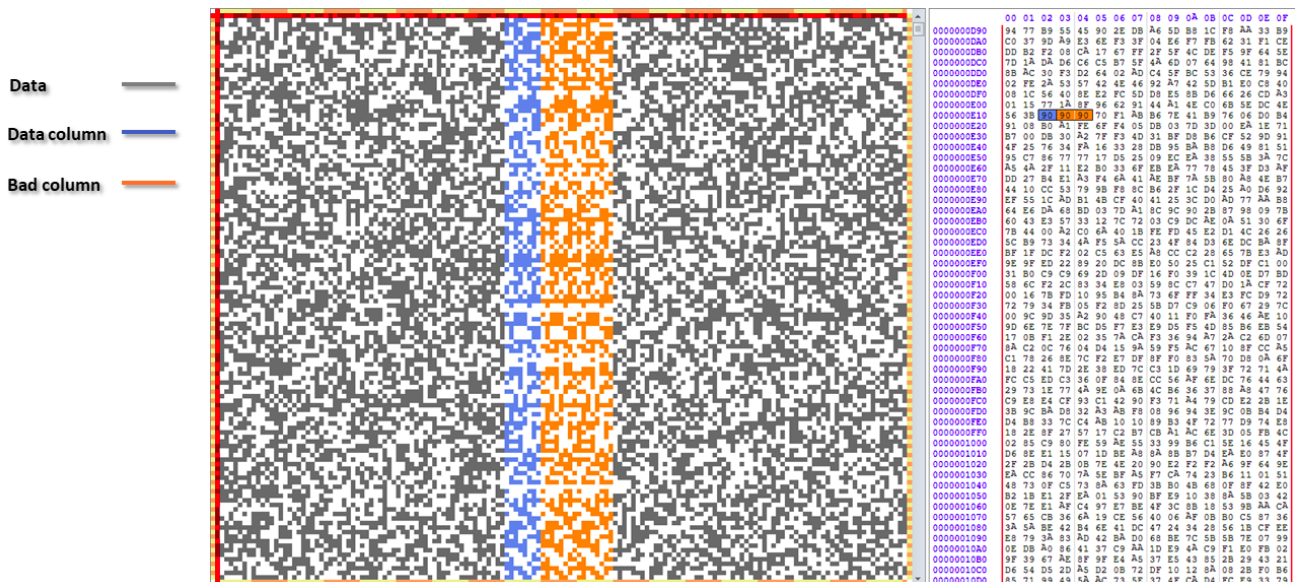
We can distinguish also the third, significantly different, type of bad column pattern, we are calling them XORed bad columns. This type of bad column is typical for Alcor Micro controllers only.

Ex.3 XOR'd - Bad column



These bad columns don't have a static pattern, like the majority of cases, but their patterns are highly dependable on recorded data. Recognition of them, via Bitmap viewer, is barely possible, since the pattern of these columns is the same as the recorded data. Similarly to other controllers, Alcor Micro fills positions of bad columns with some values, in that different that these values aren't static. These controllers are replicating bytes of data, from the buffer, onto the bad column position. This means that when the controller is composing the page in the buffer, the position, where the bad column occurs, will be filled out with the former byte from the buffer. Therefore, as a result, in Bitmap, we can spot places where several columns in a row have exactly the same hex values/pattern and these are the exact places where bad columns are present.

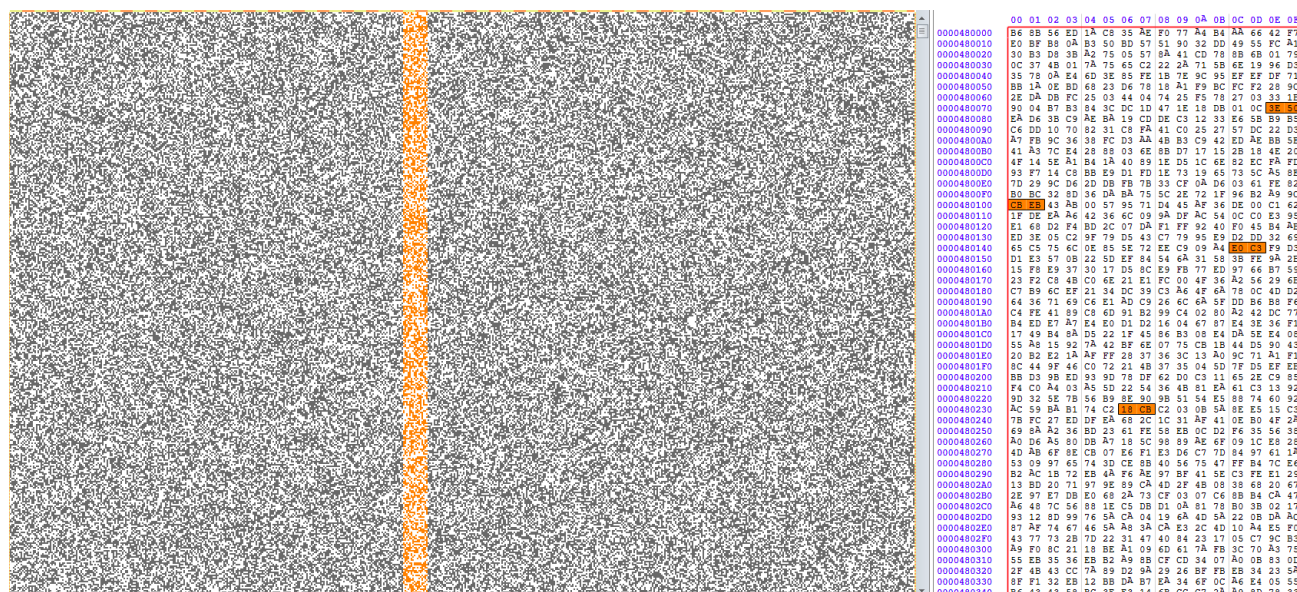
Ex.4 XOR'd - Bad column



- ① More information about XORed bad columns, their detection, and removal is widely described in this article: [Alcor Micro\(AU\) controllers - Peculiarities of data recovery.](#)

Ex.5 Invisible bad columns

There are some exceptional cases where bad columns are completely invisible through bitmap or hex viewer. Such cases are really rare, however, we've seen them several times in rebranded Phison controllers and some SiliconMotion. A simple solution for their removal isn't available, but we are offering assistance with their removal for premium support users via Help Center.

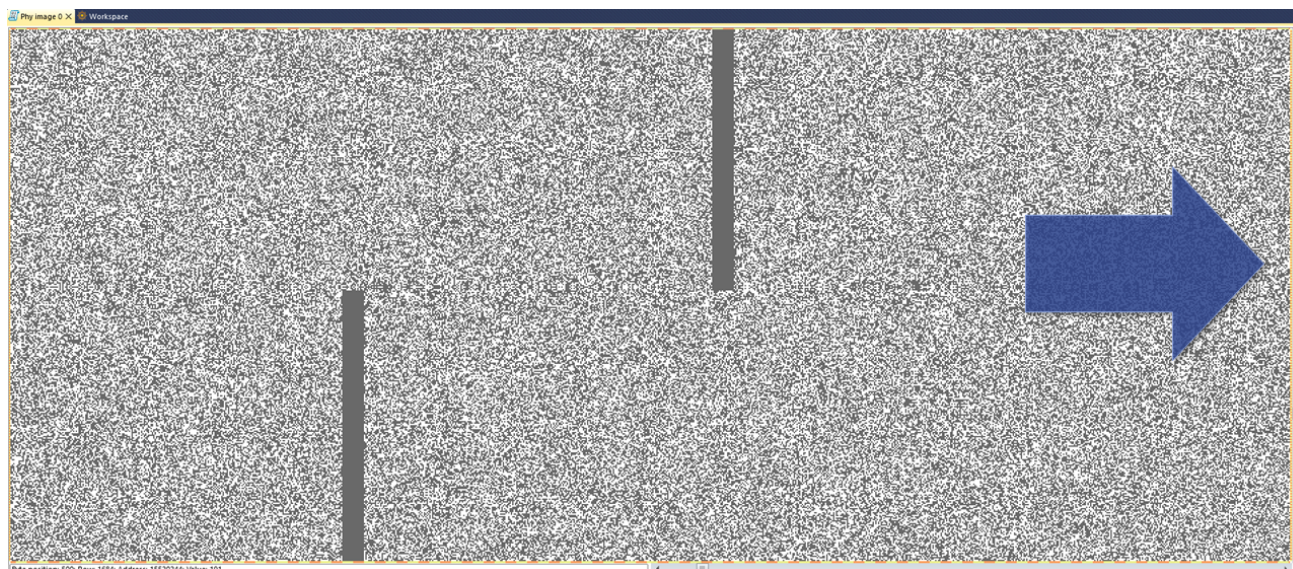


Dump quality analysis

During the chip-off data recovery, it's necessary to reverse operations/steps applied to user data by the controller. Due to that from the controller point of view, bad columns aren't visible, the controller isn't taking them into account while calculating ECC, applying an XOR key, or any other data transformation, therefore to properly work with the extracted physical image it's necessary to remove bad columns on the very beginning straight after obtaining the physical image. In the very first step after physical image extraction, it's necessary to check the presence of bad columns and there are a few very simple methods to do that.

Dump analysis

In the majority of cases, bad columns are noticeable in the dump, due to their special pattern, therefore it's possible to check if they are present by going through the bitmap of an extracted dump.



Finding ECC

To correct the dump it's necessary to find the ECC. ECC can only be detected on the dumps which don't contain bad columns, therefore it's possible to check their presence also by detecting ECC.

Find codewords

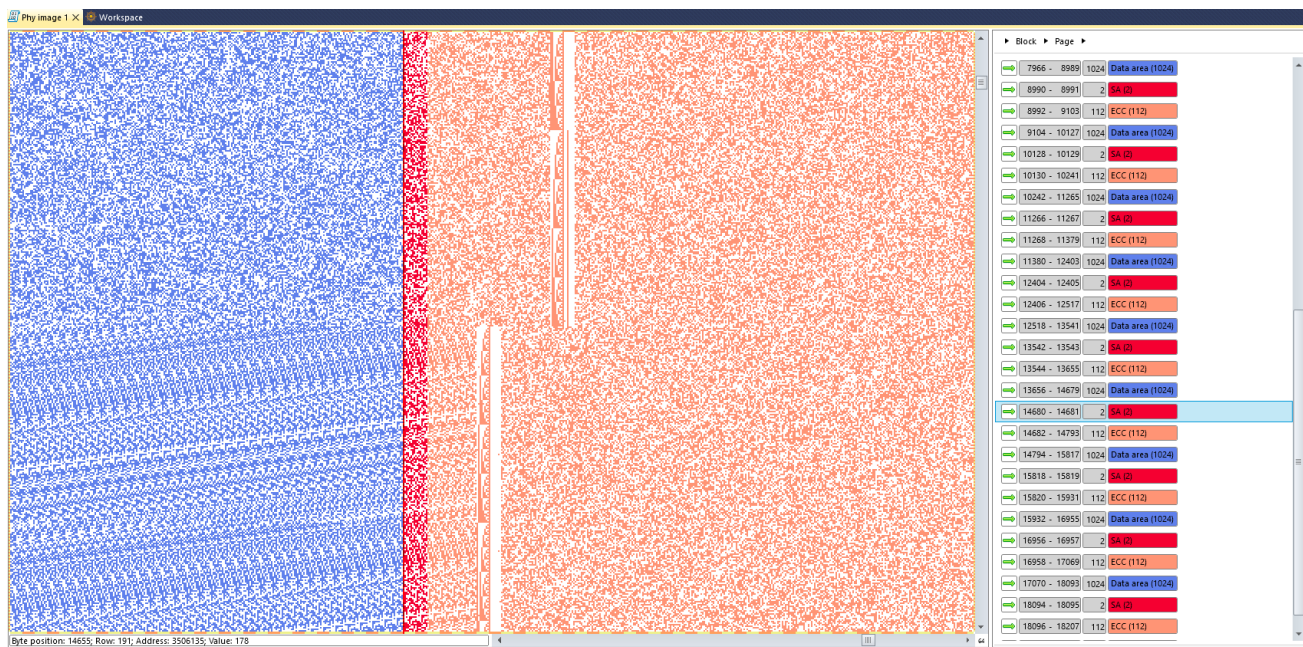
☒ All vendors
☐ Selected vendors:

Controller	Result	File
PS2251-61-5	99.00 %	PS2251-61-5_8832(ecc77)_8.bch
PS2251-67-5	99.00 %	PS2251-67-5_17664(ecc77b)_16.bch
PS2251-03-Q	12.89 %	PS2251-03-Q_17664(ecc78b)_16.bch
PS2251-09-26	12.89 %	PS2251-09-26_17760(ecc78b)_16.bch
Phison SD	0.00 %	PS2251 SD_8640(ecc44b)_8.bch
Phison_noname	0.00 %	Phison_noname_9141_8.bch
PS (monolithSD)	0.00 %	PS2251 (SD monolith)_9216(ecc106b)_8.bch
PS 2251-07-V	0.00 %	PS2251-07-V_18432(ecc120b)_16.bch
PS2231	0.00 %	PS2231_2112(ecc10b)_4.bch
PS2251 (SD monolith)	0.00 %	PS2251 (SD monolith)_17664_16.bch
PS2251-02-Q	0.00 %	PS2251-02-Q_9216(ecc120b)_8.bch
PS2251-03-Q	0.00 %	PS2251-03-Q_17664(ecc76b)_16.bch
PS2251-03-Q	0.00 %	PS2251-03-Q_9216((ecc118b+1) 8.bch

Find Select Cancel

Setting page layout

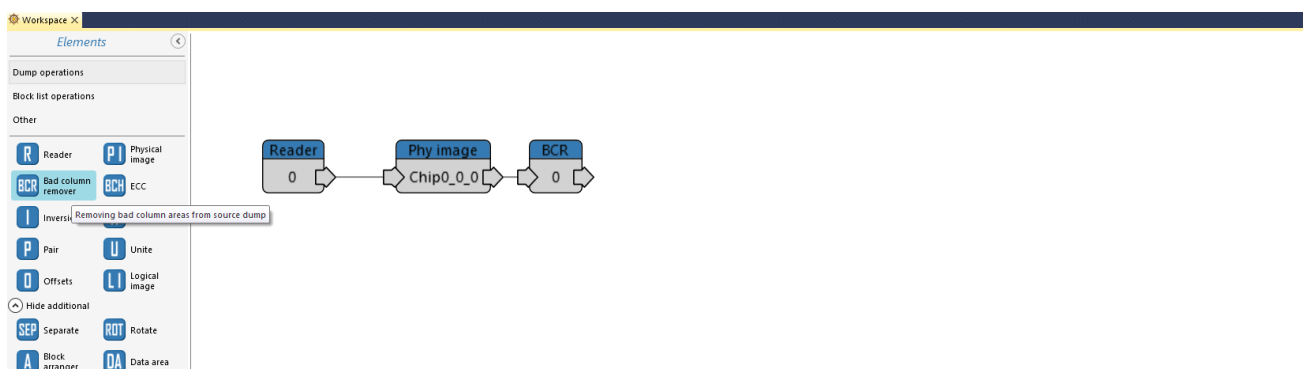
Making a page layout is an initial step of recovery and also it may help you to distinguish if bad columns are present or not, especially in case they aren't visible and ECC wasn't detected. In cases when bad columns are present, in the dump, setting page layout isn't possible since set structures will not be corresponding to structures in the dump, like in the example below.



Bad column remover - Introduction

Bad columns are defects that occur inside the physical planes of the NAND memory crystal where all user data is stored. In the same way, as the controller does, VNR reads data from memory crystals by physical pages, therefore in obtained physical images bad columns may be present as well, and in such a situation, it's necessary to remove them.

For these purposes, VNR has a special element, Bad column remover, which allows to remove all types of bad columns from physical images. BCR element disposes of semi-automatic mode with automatic detection of bad column positions and also allows to remove the bad columns manually. This tool is based on the analysis of bit column statistics. Each type of bad column has its own defined bit statistics, therefore by their analysis it's possible to detect which bytes inside the dump are bad columns.

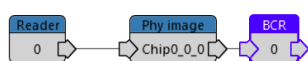


Bad column remover - Parameters

The bad column remover element has three sets of parameters, from which only the Number of Planes should be checked and configured.

These parameters are the following:

- The number of planes - Determines the number of planes inside the NAND memory crystal.
- Physical block size - **Physical size of the block** (Parameter is taken from the configuration and doesn't require to be checked.)
- Page size - **Physical page size** (Parameter is taken from the configuration and doesn't require to be checked.)



The screenshot shows the 'Parameters' window for the 'Bad column remover' element. It includes a search bar, a list of parameters, and a table for configuration.

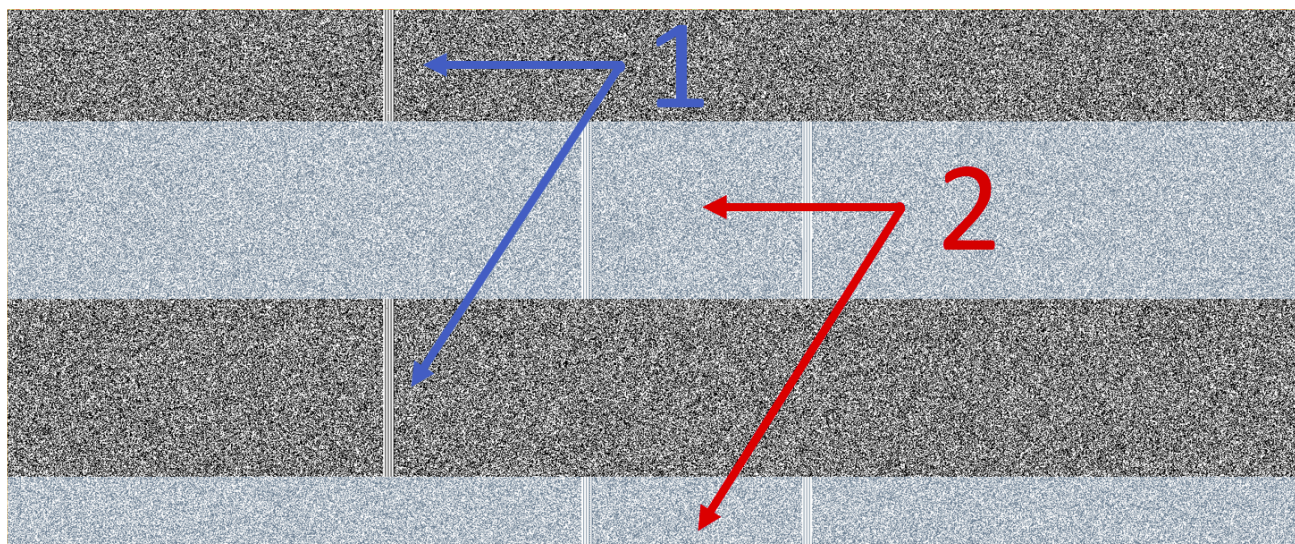
Parameters	
Enter filter string	
Element	
Name	0
Dump	
Length (bytes)	4529848320
Automatic str...	☒
Bad column remover	
Number of pl...	2 planes
Phy block size	1105920
Page size	4320
Bad columns	Edit

Number of Planes - Determination

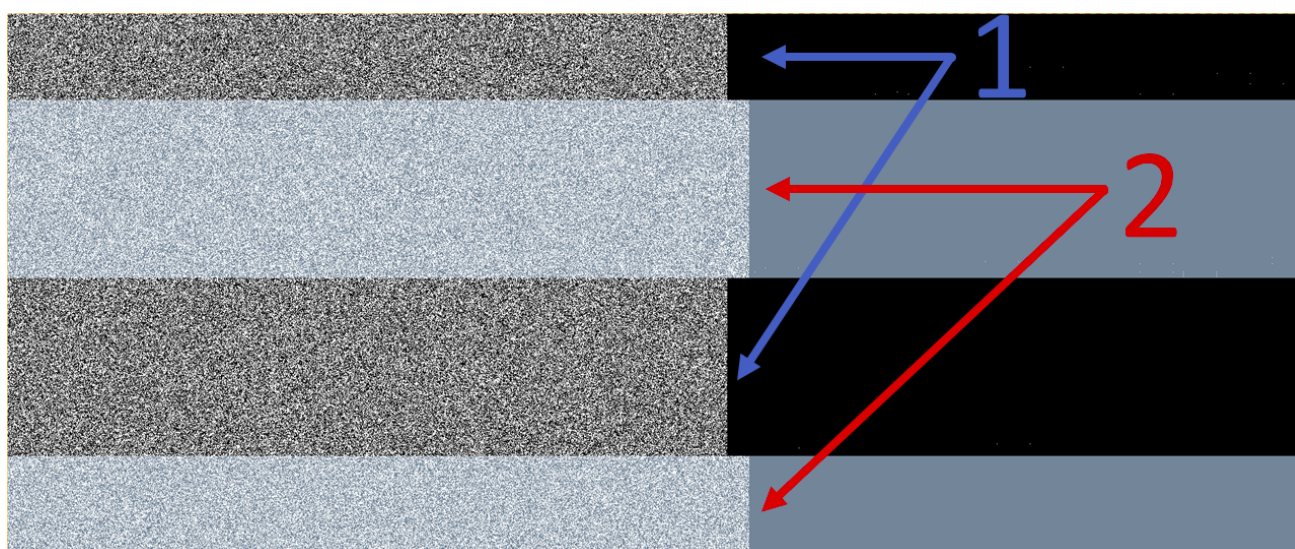
Each NAND memory crystal may have either two or multiple planes inside and for each of these units, the amount and distribution of bad columns are individual. Therefore the number of planes can be determined also by checking positions where bad columns are present. Bad columns in physical blocks from one plane will be in different positions compared to the other planes.

Ex.1

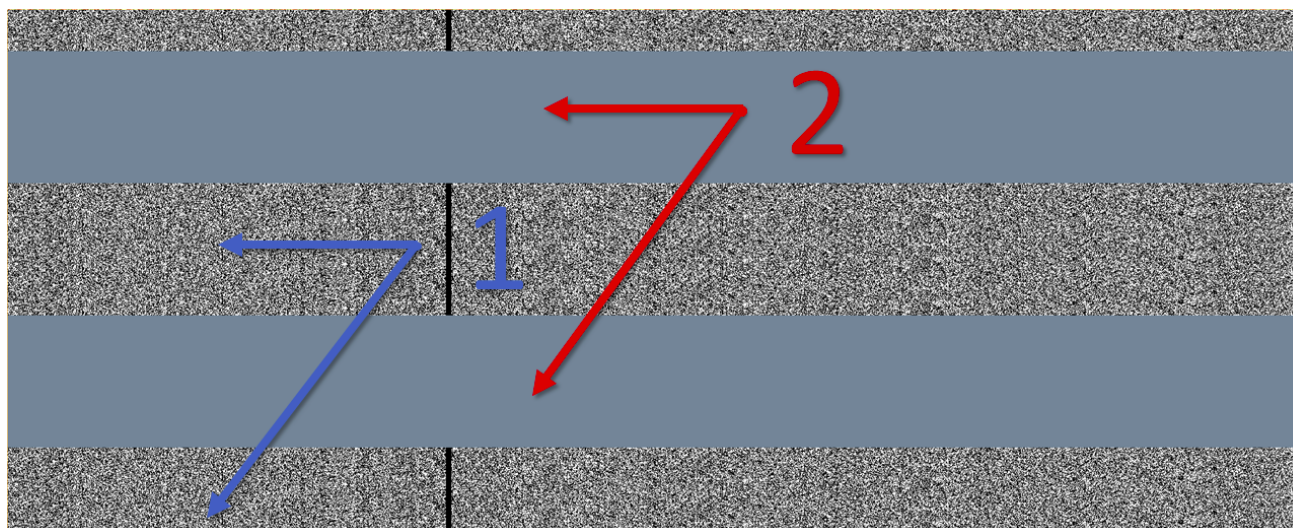
In this example, bad columns are in the same positions, every second block, which means that in this case, we have 2 Planes.

**Ex.2**

Usually, the number of bad columns inside the Planes is different, and because of that, it's also possible to determine the number of planes by checking the end of the page. This method is very useful in cases when bad columns aren't visible.

**Ex.3**

There are cases when the second plane is damaged or not used by the controller. The majority of AlcorMicro controllers for instance allocate data at first to the first plane and then to the second plane. In case a device wasn't fully written with data, the second plane may be empty, so filled only with 0xFF, like in the example below.



Bad column remover - Bad column detection and removal procedure

When the number of planes has been checked and configured in the next step positions of bad columns should be determined and marked.

Select the BCR element and click Edit, to open the environment.

Parameters

Enter filter string

Element

Name

Dump

Length (bytes)

Automatic structure ☐

Bad column remover

Number of planes

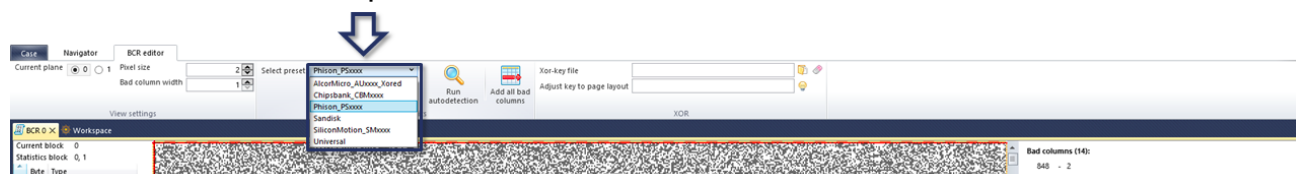
Phy block size

Page size

Bad columns

In the first step, it's necessary to choose the preset which consists of a set of rules and filters used for the bad column autodetection. Depending on the controller bad columns may have slightly different patterns/values, therefore there are presets for each popular controller vendor. The preset also determines if autodetection will proceed for each plane automatically or not. Controllers like Phison, Sandisk, or SiliconMotion are using Multi-Plane Page allocation mode which allows processing bad column analysis and removal automatically on each plane without the risk of choosing an

inappropriate block for analysis. For Chipsbank and AlcorMicro presents, autodetection is processed on Single Plane due to the Single-Plane page allocation mode used by these controllers. Single-Plane detection is also enabled for Universal preset.

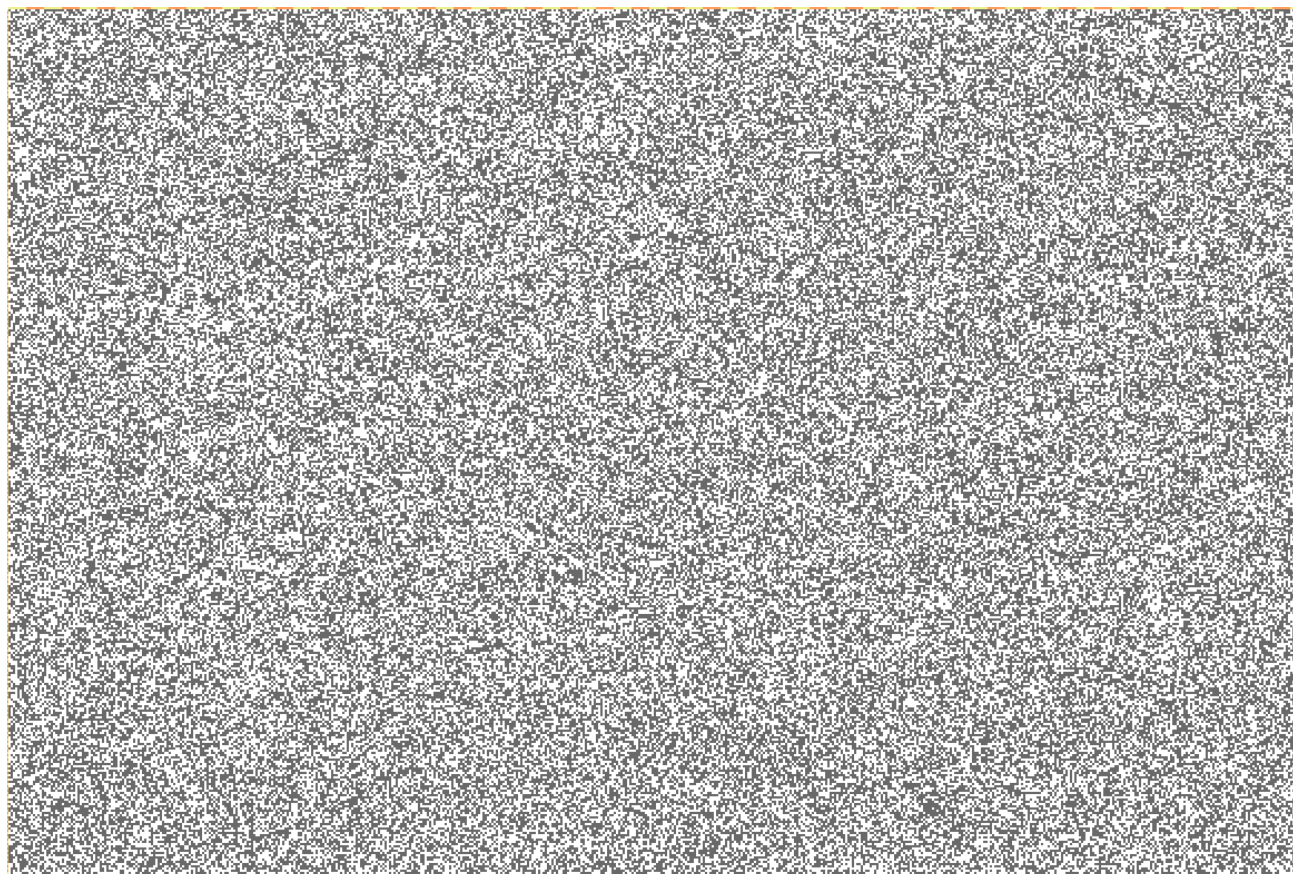


- **AlcorMicro_AUxxx_Xored** - Preset for Alcor Micro controllers with XORED bad column pattern. [Single-Plane bad column detection.](#)
- **Chipsbank_CBMxxx** - Preset for Chipsbank controllers. [Single-Plane bad column detection.](#)
- **Phison_PSxxx** - Preset for Phison controllers. [Multi-Plane bad column detection.](#)
- **Sandisk** - Preset for Sandisk controllers. [Multi-Plane bad column detection.](#)
- **SiliconMotion_SMxxx** - Preset for SiliconMotion controllers. [Multi-Plane bad column detection.](#)
- **Universal** - Universal present which consists of a set of rules for each controller (Doesn't include AlcorMicro XORED columns). Shall be selected if a controller model remains unknown. [Single-Plane bad column detection.](#)

When preset has been selected it's necessary to locate a block inside the dump viewer where byte columns will be analyzed. Due to that bad column remover operates on bit statistics it's necessary to find a block with noise/randomized pattern for analysis, in order to detect positions of bad columns.

Bad columns occur mainly inside the TLC chips, which are always XORED. As a result, most of the blocks in the dump have noise patterns and completely "randomized" data. Wherefore conditions and settings of our presets are optimized to work with exactly such blocks.

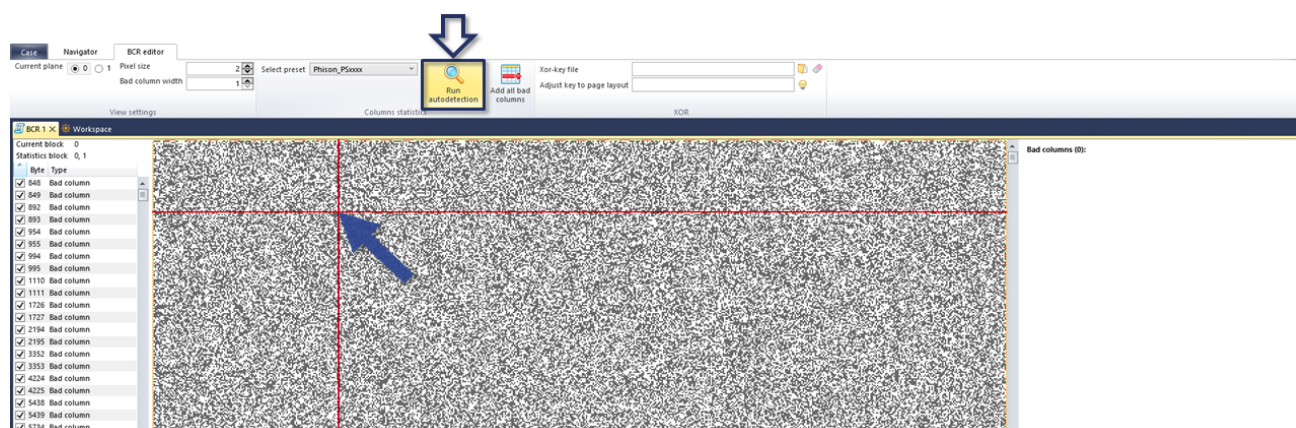
Ex. Scrambled data block



Blocks with the other patterns shall be skipped. Analysis of blocks with other patterns may result in a lot of false-positive bad columns, depending on block bit statistics.

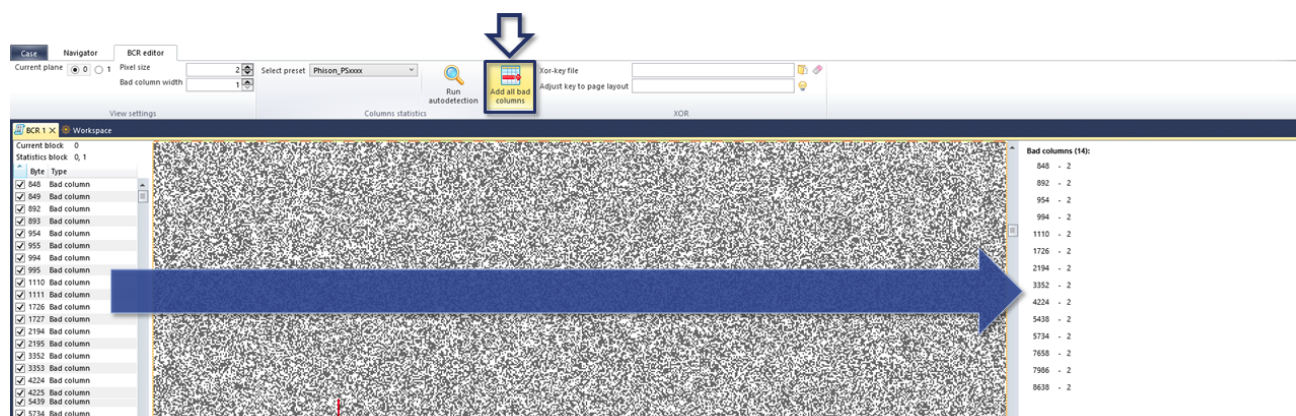
For more information about binary patterns and examples, please check the article [Binary patterns in NAND flash memory](#)

When the block has been found, select it on the bitmap viewer with LMB, and press the "Run autodetection". The tool will read statistics from the selected block and automatically compare and filter them against rules from previously selected preset. If the byte statistic will meet bad column conditions, then such byte will be classified as a bad column.

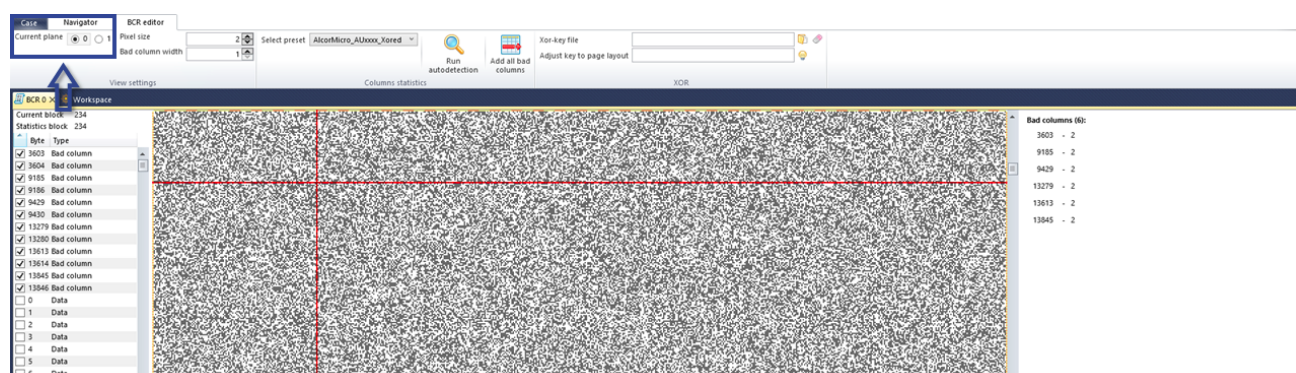


When all bad bytes have been found, it's possible to mark them as bad

columns. To do that click Add all bad columns.

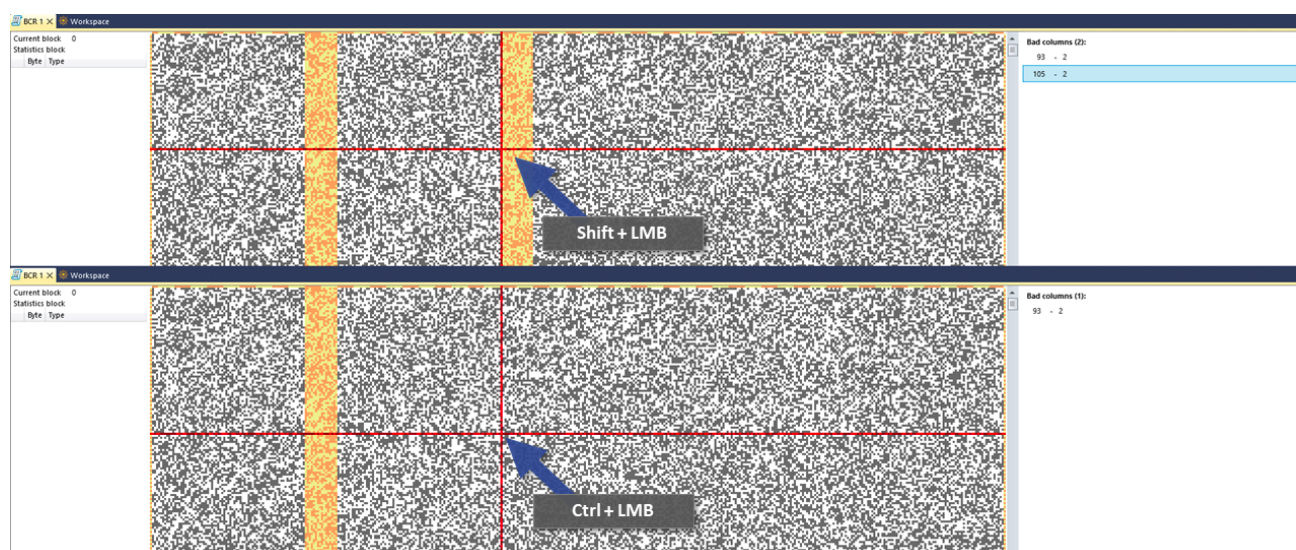


After all bad columns are marked it's necessary to repeat the removal procedure for the next planes in case that Single-Plane preset was selected. To do that switch the Current Plane from View settings.



Bad column remover - Manual bad column selection

Bad columns can be also marked manually on the bitmap viewer with a combination of Shift + LMB and unmarked with CTRL + LMB.

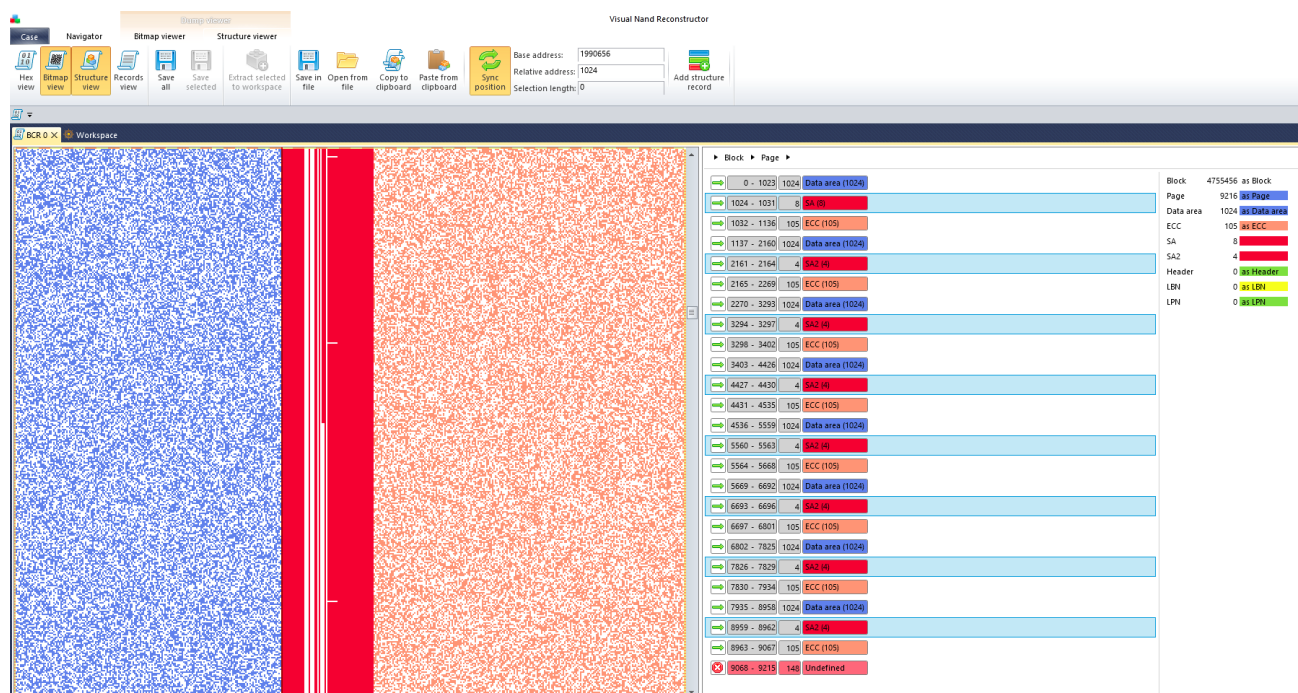


Evaluation of removed bad columns

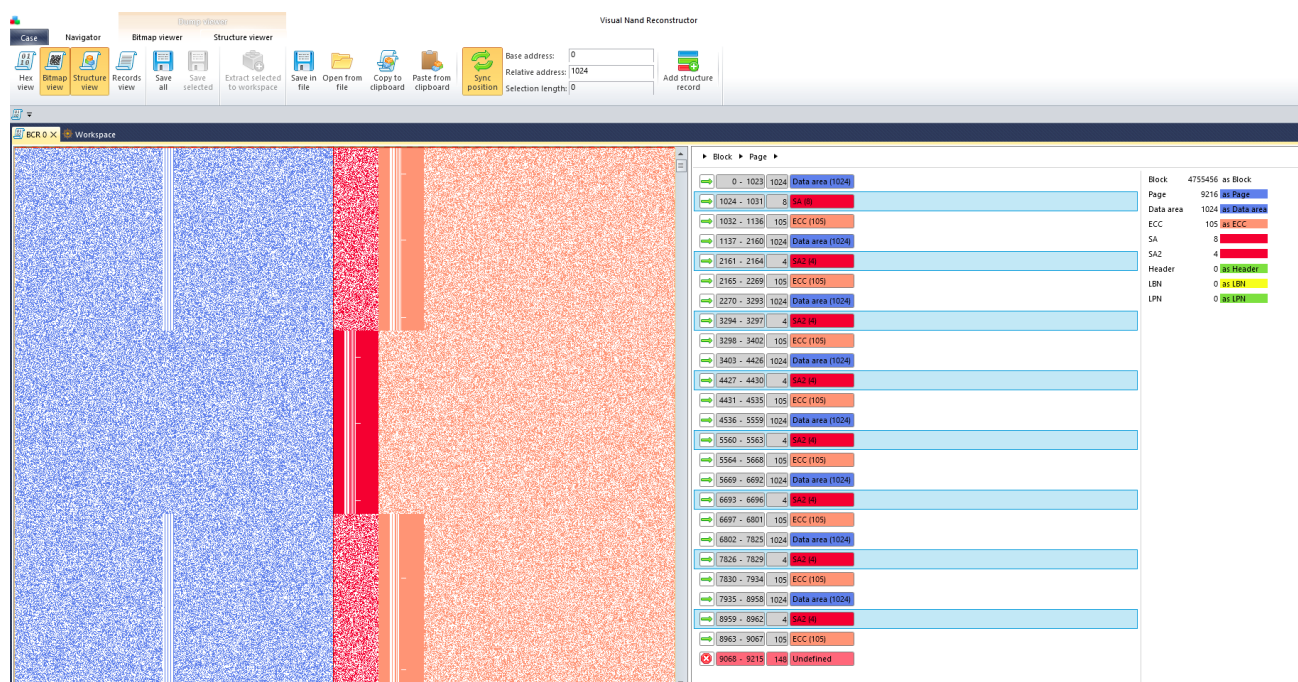
When positions of bad columns have been marked, for all planes, it's necessary to check, if all bad columns have been removed properly and there are two methods to do that.

Checking page layout

Page structures of each controller are periodic and each structure has exactly the same size, therefore it's easy to simply assign page layout and check if it corresponds visually to set structures. The easiest way is to check Services areas, in case these are available.



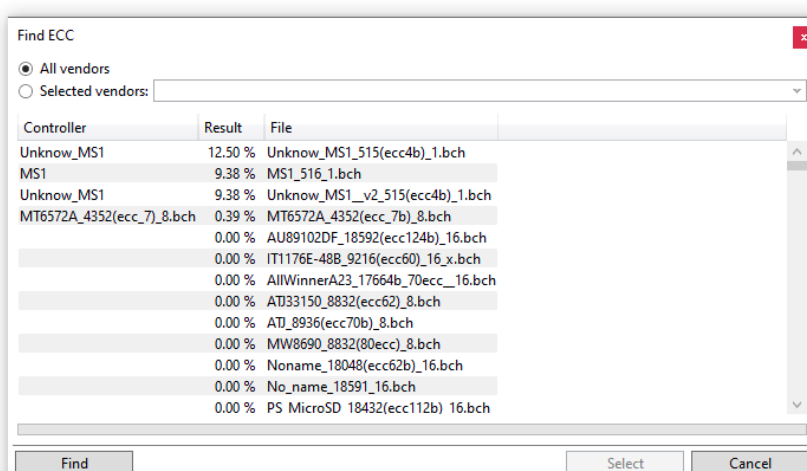
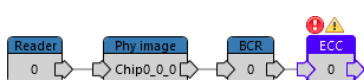
In a situation when the page layout isn't corresponding to page structures, like in the example below, bad columns shall be analyzed once again.



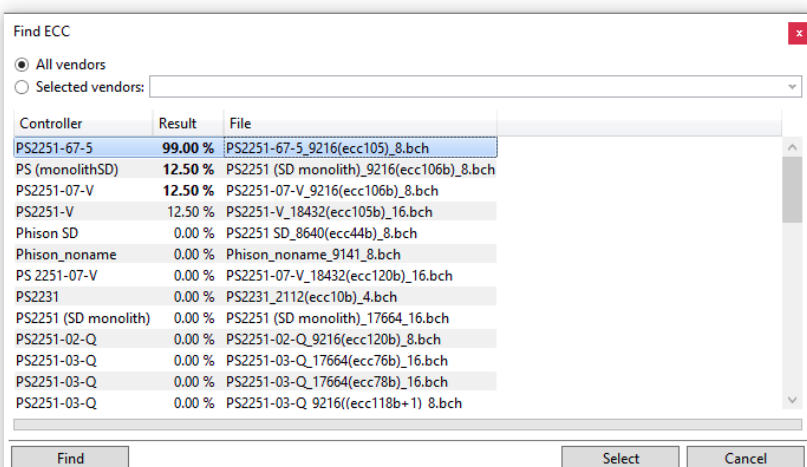
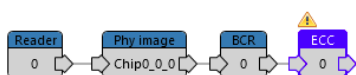
- ❗ In case when bad columns were present in multiple planes, it's necessary to check the structure in a few neighboring physical blocks.

Detection of the ECC

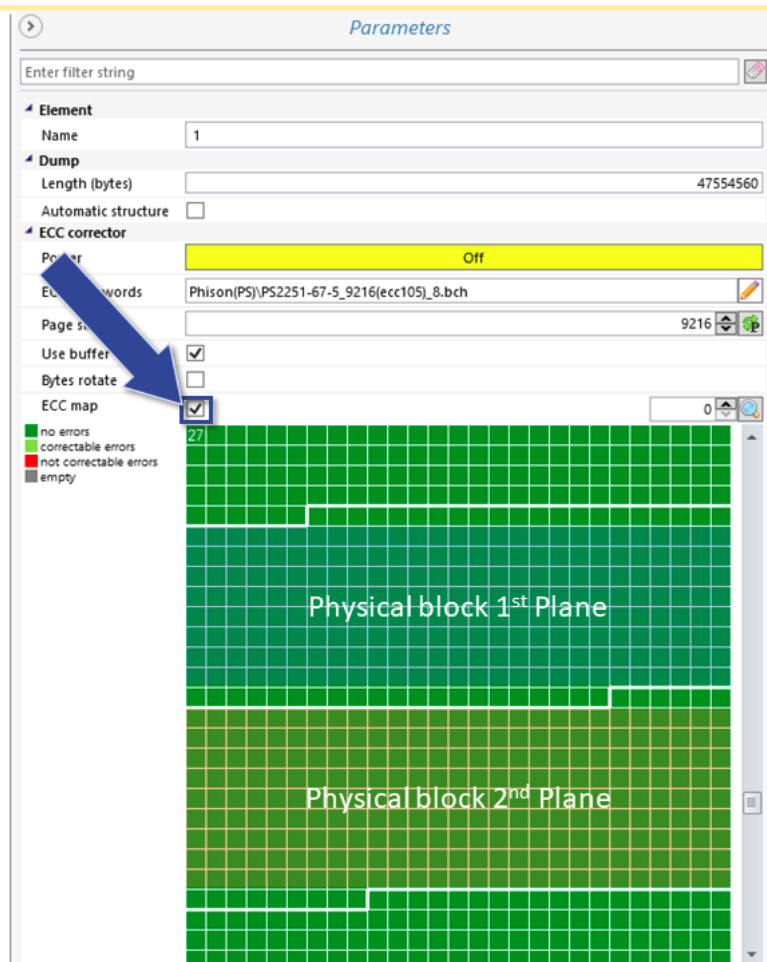
Each BCH file that is located in the VNR BCHCodewords database is designed to work with a particular page structure, therefore in case when not every bad column was removed or there are some remaining false positives, the such file will not be detected.



When the file has been detected and the ECC map shows correctable/good pages it means that ECC is working and bad columns have been removed correctly.

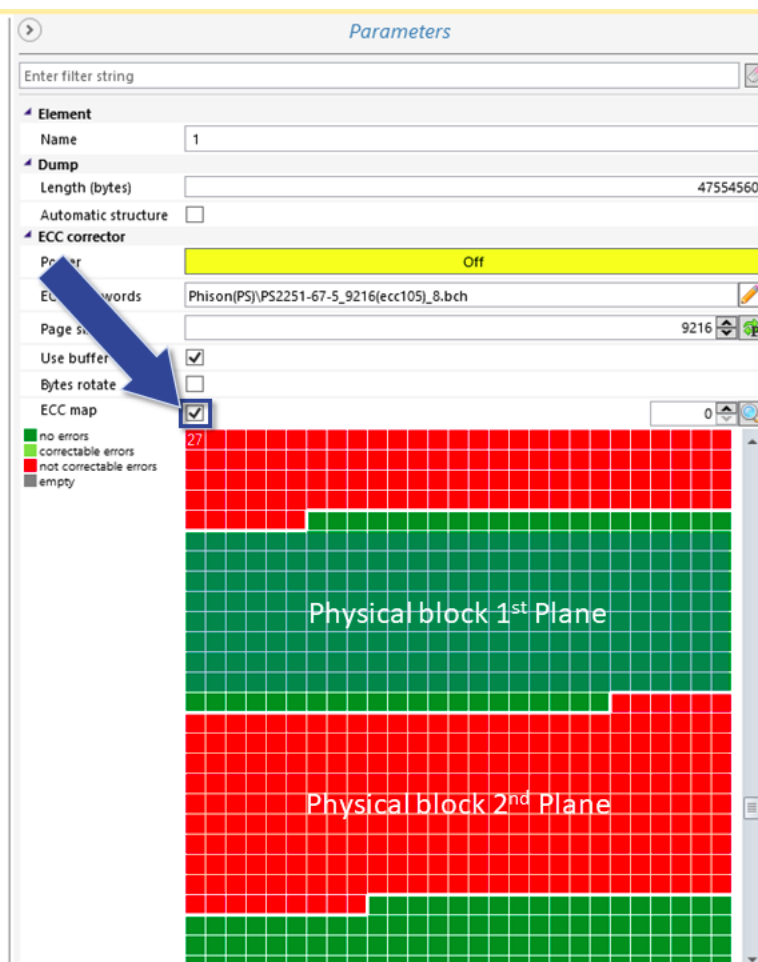


For the cases with bad columns in multiple planes, it's necessary to check if correctable pages are available in a few neighbor physical blocks in the ECC map, like one example below.



- ① The distribution of physical blocks in the ECC map is the same as on the dump viewer.

In the case when bad columns were removed incorrectly, ECC map will display only red pages for physical blocks from this exact plane.



More information about ECC design and detection is available in this article: [ECC in NAND flash memory](#).

Case studies

<https://support.rusolut.com/portal/en/kb/articles/nand-bad-column-removal>